# Кондиционеры версия 2

# Логика, которая касается пользователей.

## 03. 08. 2011

На данный момент приложение умеет содавать пользователей и назначать им роли –

|  |
| --- |
| ROLE\_USER, ROLE\_ADMIN, ROLE\_MANAGER, ROLE\_WORKER |

У каждой роли есть свои права.

|  |
| --- |
| READ, WRITE, DELETE |

|  |  |
| --- | --- |
| RoleEntity workerRole = createRole(Roles.ROLE\_WORKER.name(), Arrays.asList(  readAuthority)); | ROLE\_WORKER |
| RoleEntity adminRole = createRole(Roles.ROLE\_ADMIN.name(), Arrays.asList(readAuthority,  writeAuthority, deleteAuthority)); | ROLE\_ADMIN |
| RoleEntity managerRole = createRole(Roles.ROLE\_MANAGER.name(), Arrays.asList(readAuthority,  writeAuthority)); | ROLE\_MANAGER, |

Кроме этого, есть контроллер AdminUserController с методами –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |

Основные задачи дальше – доделать админку.

|  |  |  |
| --- | --- | --- |
| 1. изменить пользователя | ROLE\_ADMIN  ROLE\_MANAGER  ROLE\_WORKER(только себя) | Контроллер управления  Профайл |
| 1. добавить роль конкретному пользователю | ROLE\_ADMIN | Админка |
| 1. Удалить роль у конкретного пользователя | ROLE\_ADMIN | Админка |
| 1. получить пользователя | ROLE\_ADMIN  ROLE\_MANAGER  ROLE\_WORKER(только себя) | Контроллер управления  Админка  Профайл |
| 1. получить всех пользователей | ROLE\_ADMIN  ROLE\_MANAGER | Контроллер управления |
| 1. удалить пользователя | ROLE\_ADMIN | Админка |
| 1. заблокировать пользователя | ROLE\_ADMIN | Админка |

## 03. 08. 2011 - 2

На данный момент в админке созданы методы -

|  |  |  |
| --- | --- | --- |
| 1. удалить пользователя | ROLE\_ADMIN | Админка |
| 1. заблокировать пользователя | ROLE\_ADMIN | Админка |

Кроме этого, есть контроллер AdminUserController с методами –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |
| deleteUser | ROLE\_ADMIN |
| blockUser | ROLE\_ADMIN |

## 04.08.2021

На данный момент в админке есть методы –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |
| deleteUser | ROLE\_ADMIN |
| blockUser | ROLE\_ADMIN |
| addRoleToUser | ROLE\_ADMIN |
| deleteRoleFromUser | ROLE\_ADMIN |

Следующая задача – создать контроллеры для менеджера и работника

## 05.08.2021

Созданы методы обновления пользователя и получения пользователя. Оба эти метода должны быть доступным всем трем ролям, но с разницей, что точка входа для них будет разная

На данный момент функциональность реализована для менеджера и создан менеджер контроллер.

На данный момент в админке есть методы –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |
| deleteUser | ROLE\_ADMIN |
| blockUser | ROLE\_ADMIN |
| addRoleToUser | ROLE\_ADMIN |
| deleteRoleFromUser | ROLE\_ADMIN |
|  |  |

На данный момент в общем сайте для менеджеров есть методы –

|  |  |
| --- | --- |
| updateUser | ROLE\_ADMIN |
| getUserByUuid | ROLE\_ADMIN |

Следующий шаг – добавить эти же два метода (обновление пользователя и получение пользователя) для администратора

На данный момент в админке есть методы –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |
| deleteUser | ROLE\_ADMIN |
| blockUser | ROLE\_ADMIN |
| addRoleToUser | ROLE\_ADMIN |
| deleteRoleFromUser | ROLE\_ADMIN |
| updateUser | ROLE\_ADMIN |
| getUserByUuid | ROLE\_ADMIN |

На данный момент в общем сайте для менеджеров есть методы –

|  |  |
| --- | --- |
| updateUser | ROLE\_MANAGER |
| getUserByUuid | ROLE\_MANAGER |

Следующий шаг – создать метод getAllUsers для администратора и менеджера. И добавить profileController с методами обновления и получения пользователя для работника

На данный момент в админке есть методы –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |
| deleteUser | ROLE\_ADMIN |
| blockUser | ROLE\_ADMIN |
| addRoleToUser | ROLE\_ADMIN |
| deleteRoleFromUser | ROLE\_ADMIN |
| updateUser | ROLE\_ADMIN |
| getUserByUuid | ROLE\_ADMIN |
| getAllUsers | ROLE\_ADMIN |

На данный момент в общем сайте для менеджеров есть методы –

|  |  |
| --- | --- |
| updateUser | ROLE\_MANAGER |
| getUserByUuid | ROLE\_MANAGER |
| getAllUsers | ROLE\_MANAGER |

На данный момент в profile сайте для работников есть методы – эти методы разрешены пользователю исключительно в отношении себя самого

|  |  |
| --- | --- |
| updateUser | ROLE\_WORKER |
| getUserByUuid | ROLE\_WORKER |

!!! метод «Обновить пользователя» позволяет менять имя, фамилию и электронный адрес пользователя

На данном этапе эта логика закончена.

Следующий шаг – тесты и документация

### Документация

На данном этапе создается документация посредством javaDoc. Для ее создания необходимо ввести команду mvn javadoc:javadoc и документация будет создана в корневой папке проекта

### Тестирование

Готовы тесты Админ сервиса

Готовы тесты GeneralService

06.08.2021

На данном этапе все методы и контроллеры, связанные с пользователем написаны. Все методы сервиса покрыты тестами

На данный момент в админке есть методы –

|  |  |
| --- | --- |
| createUser | ROLE\_ADMIN |
| deleteUser | ROLE\_ADMIN |
| blockUser | ROLE\_ADMIN |
| addRoleToUser | ROLE\_ADMIN |
| deleteRoleFromUser | ROLE\_ADMIN |
| updateUser | ROLE\_ADMIN |
| getUserByUuid | ROLE\_ADMIN |
| getAllUsers | ROLE\_ADMIN |

На данный момент в общем сайте для менеджеров есть методы –

|  |  |
| --- | --- |
| updateUser | ROLE\_MANAGER |
| getUserByUuid | ROLE\_MANAGER |
| getAllUsers | ROLE\_MANAGER |

На данный момент в profile сайте для работников есть методы – эти методы разрешены пользователю исключительно в отношении себя самого

|  |  |
| --- | --- |
| updateUser | ROLE\_WORKER |
| getUserByUuid | ROLE\_WORKER |

Следующая задача – подключить Spring Security так, что бы открыть доступ к конкретным методам только тем, кто конкретно имеет к ним доступ.

Ветка -> Security

После того, как мы добавляем зависимость в pom - при запросе из постмена мы получаем ошибку 401 - Unauthorized. Пока все работает как нужно

После добавления метода

@Override  
protected void configure(HttpSecurity http) throws Exception {  
 http.csrf().disable()  
 .authorizeRequests()  
 .anyRequest().authenticated();  
  
 http.headers().frameOptions().disable();  
}

В класс

public class WebSecurity extends WebSecurityConfigurerAdapter

мы должны в постмене получить уже 403 ошибку.

Эта ошибка говорит о том, что нам необходимо сначала пройти аутентификацию (то есть доказать, что у нас есть право доступа к этому методу) и потом только контроллер нас в него пропустит.

Для того, что бы пройти – нам необходим токен.

Добавляем создание и обработку токена

### Идея токена

Идея в том, что при логине пользователь получит свой токен, по которому его будет определять юзер сервис и давать ему доступ ко всему

### Создание токена

Создаем класс LoginRequestDto. Именно объект этого класса должен отправить пользователь при «логине»